AN APPLICATION OF KALMAN FILTERING TO UNDERWATER TRACKING

Eric James Benson

NAVAL POSTGRADUATE SCHOOL
Monterey, California

THESIS

AN APPLICATION OF KALMAN FILTERING TO
UNDERWATER TRACKING

by

Eric James Benson

December 1976

Thesis Advisor: H. A. Titus

Approved for public release; distribution unlimited.

REPORT DOCUMENTATION PAGE

READ INSTRUCTIONS BEFORE COMPLETING FORM

1. REPORT NUMBER

2. GOVT ACCESSION NO.

3. RECIPIENT'S CATALOG NUMBER

4. TITLE (and Subtitle)
An Application of Kalman Filtering to Underwater Tracking

5. TYPE OF REPORT & PERIOD COVERED
Master's Thesis
December 1976

6. PERFORMING ORG. REPORT NUMBER

7. AUTHOR(s)
Eric James Benson

8. CONTRACT OR GRANT NUMBER(s)

9. PERFORMING ORGANIZATION NAME AND ADDRESS
Naval Postgraduate School
Monterey, California 93940

10. PROGRAM ELEMENT, PROJECT, TASK AREA & WORK UNIT NUMBERS

11. CONTROLLING OFFICE NAME AND ADDRESS
Naval Postgraduate School
Monterey, California 93940

12. REPORT DATE
December 1976

13. NUMBER OF PAGES

14. MONITORING AGENCY NAME & ADDRESS(if different from Controlling Office)
Naval Postgraduate School
Monterey, California 93940

15. SECURITY CLASS. (of this report)
Unclassified

15a. DECLASSIFICATION/DOWNGRADING SCHEDULE

16. DISTRIBUTION STATEMENT (of this Report)
Approved for public release; distribution unlimited.

17. DISTRIBUTION STATEMENT (of the abstract entered in Block 20, if different from Report)

18. SUPPLEMENTARY NOTES

19. KEY WORDS (Continue on reverse side if necessary and identify by block number)

20. ABSTRACT (Continue on reverse side if necessary and identify by block number)

A program was developed to improve the on-line measurement capability of the three-dimensional , underwater tracking ranges at the Naval Torpedo Station, Keyport, Washington. The program utilizes a Kalman Filter routine to minimize the effects of measurement noise in determining the true target position. The gain schedule used by the filter is calculated off-line and may be varied based on the tracking requirements. Listings of both of the Fortran programs are included.

Simulated exercises were run utilizing a variety of gain schedules. Results of these simulations will assist NTS engineers in the implementation and operation of the program using the NTS computer facility. Details of the simulation procedure and a listing of the track generator program are included.

AN APPLICATION OF KALMAN FILTERING TO UNDERWATER TRACKING

by

Eric James Benson
Lieutenant, United States Navy
B.S., United States Naval Academy, 1971

Submitted in partial fulfillment of the
requirements for the degree of

MASTER OF SCIENCE IN APPLIED SCIENCE

from the

NAVAL POSTGRADUATE SCHOOL
December 1976

ABSTRACT

A program was developed to improve the on-line measurement capability of the three-dimensional, underwater tracking ranges at the Naval Torpedo Station, Keyport, Washington. The program utilizes a Kalman filter routine to minimize the effects of measurement noise in determining the true target position. The gain schedule used by the filter is calculated off-line and may be varied based on tracking requirements. Listings of both of the Fortran programs are included.

Simulated exercises were run utilizing a variety of gain schedules. Results of these simulations will assist NTS engineers in the implementation and operation of the program using the NTS computer facility. Details of the simulation procedure and a listing of the track generator program are included.

TABLE OF CONTENTS

(This page intentionally blank)

I. INTRODUCTION

The Naval Torpedo Station, Keyport, Washington currently operates two three-dimensional underwater tracking ranges with the capability of accoustically tracking torpedoes and similar water-borne targets. The computer system in use provides a printed record and plotted display of the measured track. As with most tracking systems the measured data is corrupted by noise. This noise is due to the combined effects of environmental factors and the measurement instruments.

These noisy tracks are later analyzed to remove those measurements judged to be most inaccurate on the basis of total track statistics. Thus, it is only at some later time that a smooth representation of the track is available.

In the near future an updated computer system will be brought on-line. The necessity of designing new programs provides the opportunity for expanding the real-time capability of the system. It is desired that a smoother track be available in real-time without the loss of any of the measured data. A long range prediction capability and the ability to handle various data rates is also desirable.

The above requirements and the ready access to the noisy measurements of target position indicate the applicability of a Kalman filter routine to improve the real-time capability of the three-dimensional tracking ranges at NTS.

II. THEORY

The development and use of the equations used in the Kalman filter have been widely documented. Rather than include the derivation here the reader is referred to [1,2] or similar works.

In this presentation the equations used will be listed with a brief description of each. The system is assumed to be linear and free of deterministic forcing inputs.

We characterize the state by the following state difference equations

$$X(k+1) = PHI * X(k) + GAMMA * W(k)$$

and noisy measurement equations

$$Z(k) = C * X(k) + V(k)$$

where

X(k) is the n-dimensional state vector at time K

Z(k) is the m-dimensional measurement vector at time K

V(k) is the m-dimensional random noise vector at time

W(k) is the m-dimensional random forcing input at time K, and

PHI and C are constant matrices.

The estimator equations are given by

$$\hat{X}(k|k) = \hat{X}(k|k-1) + G(k)*[Z(k) - C * \hat{X}(k|k-1)]$$

and

$$\hat{X}(k|k-1) = PHI * \hat{X}(k-1|k-1),$$

where

$\hat{X}(k|k)$ is the estimate at time K given K

measurements

$\hat{X}(k|k-1)$ is the estimate at time K given K-1 measurements, and

$G(k)$ is the Kalman filter gain at time K.

The estimator is initialized by setting $X(0|-1)$ equal to the mean of the random target entry points.

The gain (G) and theoretical covariance of error (P) equations are given by

$$G(k) = P(k|k-1) * C^T * [C*P(k|k-1)*C^T + R(k)]^{-1}$$

$$P(k|k) = [I - G(k)*C]*P(k|k-1)$$

$$P(k+1|k) = PHI*P(k|k)*PHI^T + Q$$

where

$R(k)$ is the covariance of the measurement noise,

$P(k|k)$ is the theoretical covariance of the estimation error, and

Q is found by

$$Q = GAMMA*cov(W)*GAMMA^T$$

where cov(W) is the covariance of the forcing input. In this application the measurement noise causes fluctuations in the velocity estimate suggesting that there is an acceleration input. This "acceleration" is treated as a random forcing input.

GAMMA is given by

$$\begin{bmatrix} T^2/2 \\ \\ T \end{bmatrix}$$

These equations are initialized by setting $P(0|-1)$ equal to the variance of the intiial state estimate.

III. PROBLEM DEFINITION

The design and formulation of a new tracking program may best be defined in terms of the present system and the improvements to be made. Details of the reduction of the accoustically measured track into the usable data format of XYZ positioning on the ranges are included in [3]. It is assumed that all data has been reduced to the three-dimensional coordinates for convenient use as track input to the program.

Under the present system, data is received and stored on magnetic tape for later smoothing. simultaneously, the measured position of the target is printed with other pertinent data (time, point count, array number, etc.). Two plots are also generated: target position is superimposed on a range chart in XY coordinates; and, target depth is plotted in XZ coordinates.

The typical torpedo track will have two modes: search and pursuit of its target. The search mode consists of a helical track that does not lend itself to linear approximation. Because this portion of the track is of minor importance to range observers, any inaccuracies caused by a linear approximation can be tolerated. The pursuit mode will be a constant velocity track which can easily be modeled by a linear, time-invariant system.

The Kalman filter (see Figure 1) provides an accurate discrete approximation to the continuous target track.

The current smoothing techniques (post-run analysis)

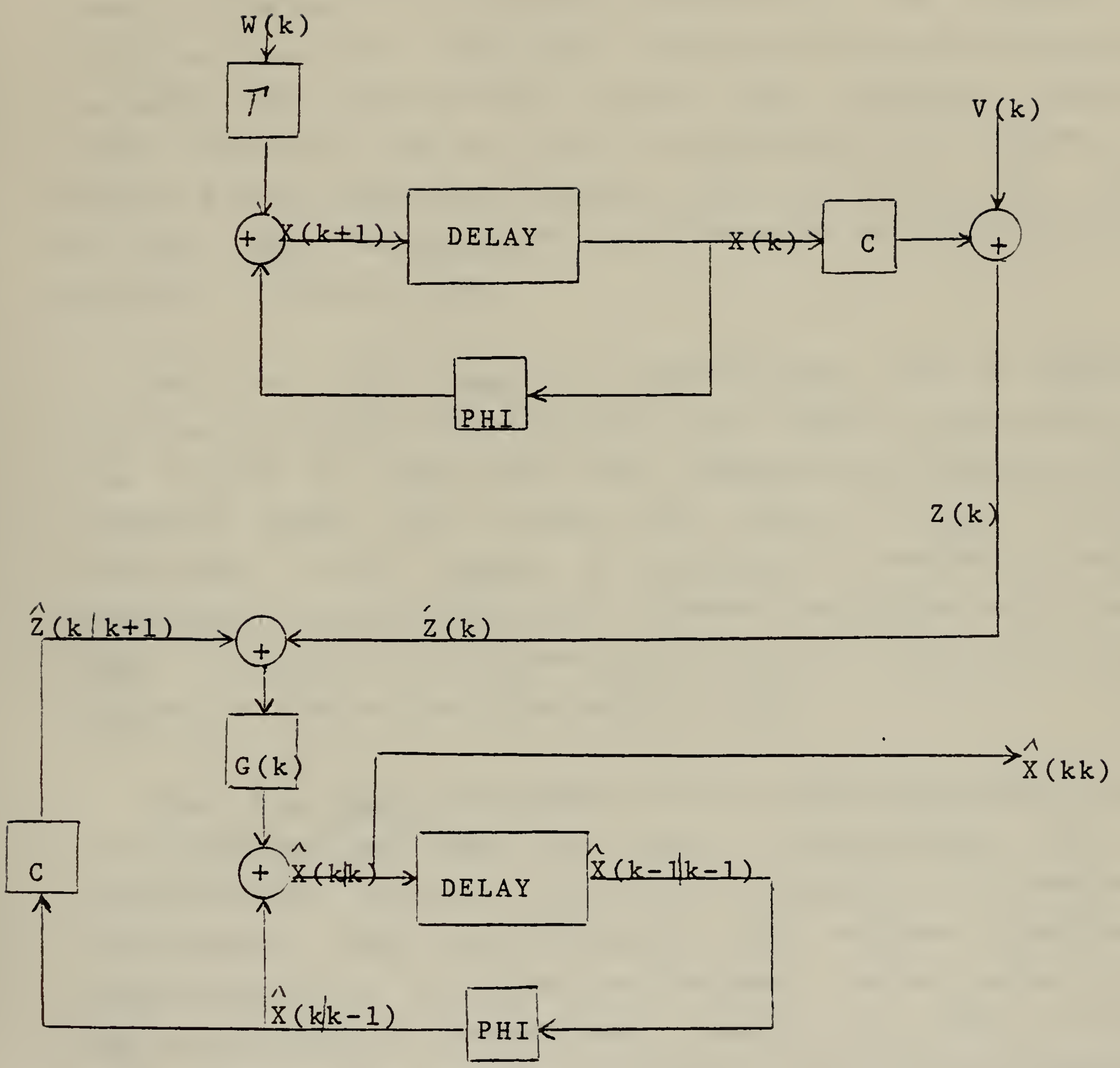

Figure 1 - BLOCK DIAGRAM OF THE DISCRETE KALMAN FILTER

remove those points that are the most corrupted by noise. This type of routine initially considers all of the measured track points. The position at any time is then estimated based on all of the measurements. The resultant track is an error minimizing curve fitted to the complete measured track. The use of the Kalman filter will not affect this type of post-run analysis, as all of the measurements are retained. The filter will, in real time, provide a smoother track through the use of the predictor and weighted (the Kalman gain) correction factor. Thus, by using the filter we are able to overcome the problem of noisy measurements without the loss of data.

Many of the required computations can be completed off-line so as not to jeopardize the timely application of the filter to the track data. Because the geometry of the tracking range will remain the same, we can utilize previously run tracks to develop a reliable set of measurement noise statistics. These statistics will then be used to compute the Kalman gain schedule, which may then be stored for use at the appropriate time.

Use of a priori knowledge of the noise statistics allows the programmer and user not only to precompute the gain schedule but also to test the program via simulation. Variations, such as allowances for various degrees of maneuverability, may also be compared. We are able to base the assumptions of measurement errors on noise criteria only, because the assumed location of the array hydrophones has been shown to be highly accurate. [4]

Two general assumptions have been made about the noise in the application of the filter. First, it is assumed that the measurement noise is random with a mean of zero and has a Gaussian distribution. Also, the noise has been taken to be independent of the distance between the target and the

array sensors. This assumption is made with consideration given to the ray path reconstruction routine used.

IV. PROGRAM DESCRIPTIONS

A. THE KALMAN FILTER

The Kalman filter program has been designed to provide an improved tracking routine using a minimum of computer storage and computation time. The filter routine has the capability cf processing a variable data rate limited only by the execution time of the routine and the accuracy desired (in the discrete approximation). This data rate is selected prior to each run, as is the advance prediction feature.

In addition to the normal one-step-ahead prediction of the filter, the user may select a long range predictor prior to each track run. This feature will enable observers to evaluate qualitatively parameters such as torpedo homing capability during the test run. The long range predictor simply advances the current filtered position the desired number of sample intervals. The current filtered velocity estimate is used in this prediction.

The operation of the filter may adversely be affected by large scale measurement errors. Range operations experience indicates that measurement errors on the order of 10^3 feet are occassionally recorded. One error of this magnitude would invalidate the filtered output for many subsequent sample intervals. Because of the feedback operation of the filter there must be protection against this form of catastrophic failure.

This protection is provided by establishing limits of acceptability about each of the measurements. The range of this acceptability "gate" is adjustable and may be set by the user prior to each exercise run. (See Appendix B on program requirements for a further explanation.)

Measurements that fall outside the gate are regarded as unacceptable; the gain schedule is set to zero for that measurement and the filter is advanced based on prior estimates.

The program is liberally documented through the use of "comment" cards. These comments will enable any user to operate the program with a minimum of supportive documentation. It is intended that only a basic knowledge of Kalman filtering and Fortran programming are necessary to operate the program. Ease of implementation has been given a high degree of consideration in the design of the program.

B. THE GAIN SCHEDULE

The Kalman gain schedule is calculated using a program independent of the filter routine. This separate program permits prior off-line computation and storage of the gain schedule. For a given tracking run the optimal gain schedule may be computed from the matrix inputs to the routine. These gains are then stored for use during the actual tracking exercise. If the user chooses to alter the gain schedule used, the independence of the gain program from the filter provides flexibility.

In operating the filter, any gain schedule (optimal or sub-optimal) may be used. For example, a higher gain value applied to each measurement would allow for a higher degree

of target maneuverability, but would provide less smoothing.

This program is completely self-supportive; it requires no additional subprograms or routines.

C. TRACK SIMULATOR

A track simulator program is also included. Its inclusion is not necessary to operate the Kalman filter routine at NTS. It is included here only to acquaint NTS engineers with the simulation technique used in program testing and verification.

The track generator uses a random number generator to select a track origin. The track is then advanced from that point based on user selected X Y and Z velocities. The actual track is then corrupted by noise to provide noisy position measurements.

The actual track and the measured, noisy positions are available as program output.

This program uses the subroutine SNORM, an IBM-360 library subroutine. SNORM is a random number generator that produces a normal Gaussian distribution.

V. SIMULATION PROCEDURE

The Kalman filter program was designed to handle efficiently (in terms of computer usage) the tracking measurements and to provide the extra features desired by NTS range observers. Once completed it was necessary to verify accurate operation of the program in an NTS range-type environment. One means of verification was to model the working environment and to conduct tracking exercise simulation using the model.

Several simplifying assumptions have been made in the model. Simplification was made only where it was determined that no conflict existed with realistic operational parameters. The primary reason for making these changes was to increase the readability of the printed and plotted output data.

In the test model, the target track was generated using velocity along the range baseline only. Originally the track was modeled with principal motion along the range baseline (X-axis) and a slight crossing trajectory (Y-axis). By eliminating the Y-component of velocity the magnitude of the noise corruption is represented as the full range of the absissa on the XY position plot.

The program was tested using a track that incorporated velocity along each of the coordinate axis but analysis of the filtered output data showed no change in the performance of the filter.

Various target velocities were used for program

verification. No differences in performance were noted over the velocity range from twenty to fifty-five knots. The final model employs a target moving at forty knots (a displacement of ninety feet between measurements).

Two considerations were made in computing the gain schedule used in testing. Statistical data obtained from previous range tracking exercises was used to initialize the gain computations. Additionally the Q matrix was varied to test the the degree of manuverability that could be handled by the filter.

No attempt was made to test directly the "gate" feature of the filter. However, measurements greatly corrupted by noise were noted in each of the simulations, providing a test of the filter's ability to reject highly erroneous measurements.

No attempt has been made to distinguish which particular array supplied the data. It was noted in the problem definition that input data to the filter was assumed to be in three-dimensional range coordinates. It has been assumed that current methods of calculating three-dimensional data will be retained and will provide measurement input to the filter in terms of the range baseline coordinate system. Based on this assumption there is no need to determine which array supplied the data.

The primary purpose of the simulation has been to test operation of the filter in such a manner that it might be implemented at NTS, Keyport with only minor changes. Attempts have been made to anticipate and to resolve any problems that might arise.

VI. SIMULATION RESULTS

After the Kalman filter program had been designed and checked for Fortran and logic errors, it was necessary to test it in an environment similar to that seen in actual range operations. Computer modeled simulations provided a rapid, lcw-cost method for verifying the accuracy of the program. Several tests were run using a variety of operational parameters, as explained below.

Most of the measured data from the NTS ranges is received at about 1.3 second intervals. The few remaining exercises utilize a data rate one-half the above. These data rates establish the absolute limits of any processing routine. One of the important aspects of the simulation is the cycle time of the filter.

All of the simulation exercises were conducted using the IBM-360 system at the Naval Postgraduate School under a time-shared mode. The tests were conducted at various times of the day when system user load ranged from moderate to heavy.

The average time of operation of the filter was 1.4240 hundredths-of-a-second. Given this wide margin of acceptability, it is assumed that no difficulty will arise when the filter is operated on the NTS range computer system.

In the Kalman gain schedule, the steady-state values were achieved after seven to twelve sample intervals depending on the input values assigned to the Q matrix. The

gain schedule shown in Table 1 is based on a target that is expected to undergo little or no maneuvers (accelerations of less than 3.2 feet/second^2). For all measurements after the last indicated time the gain values are equal to the last value listed.

TIME	POSITION GAIN	VELOCITY GAIN
1	0.990099	0.0
2	0.994248	0.754539
3	0.835858	0.398822
4	0.725189	0.284304
5	0.669428	0.252953
6	0.649976	0.248808
7	0.646037	0.249742
8	0.645805	0.250140
9	0.645766	0.250005
10	0.645621	0.249853
11	0.645519	0.249798
12	0.645481	0.249792
13	0.645474	0.249794
14	0.645474	0.249794
15	0.645474	0.249794

Table 1 - GAIN SCHEDULE FOR A NON-MANEUVERING TARGET

Two other gain schedules were used in the simulation and are shown in Tables 2 and 3. These schedules assume target accelerations of 10 feet/second^2 and 20 feet/second^2, respectively. Again the last values shown are the steady-state values and are to be applied to all subsequent measurements.

A. NON-MANEUVERING TARGET

When the target is in a non-maneuvering mode the only fluctuations from its track will be due to noise. Because this noise corruption is undesirable, the gain schedule used should provide for maximum smoothing of the target track.

The gain schedule (Table 1) was calculated assuming that the initial position estimate could be determined within ±100 feet of the actual position. By assuming that the variance due to the noise would be approximately three feet/second^2, maximum smoothing with an allowance for slight irregular target accelerations was achieved.

Using these gains with a simulated noisy track, the filtered position estimates were statistically compared with the measured positions and the known true track. The deviation of the filtered track points from the true track was approximately one-half that displayed by the measured positions. The standard deviation of the filtered track estimates about the true track was approximately equal to the theoretical value predicted in the calculation of the gain schedule. This favorable comparison of experimental and theoretical values indicates that proper initial values have been used in calculating the Kalman gain schedule.

TIME	POSITION GAIN	VELOCITY GAIN
1	0.990099	0.0
2	0.994273	0.757905
3	0.869084	0.548191
4	0.843011	0.553351
5	0.842683	0.555230
6	0.842387	0.554583
7	0.842279	0.554577
8	0.842276	0.554587
9	0.842275	0.554585
10	0.842275	0.554585
11	0.842275	0.554585

Table 2 - GAIN SCHEDULE FOR A MANEUVERING TARGET (APPROXIMATELY 1/3G ACCELERATION)

TIME	POSITION GAIN	VELOCITY GAIN
1	0.99099	0.0
2	0.994342	0.767076
3	0.915940	0.759631
4	0.915308	0.765307
5	0.914846	0.764878
6	0.914830	0.764944
7	0.914828	0.764940
8	0.914828	0.764940
9	0.914828	0.764940

Table 3 - GAIN SCHEDULE FOR A HIGHLY MANEUVERING TARGET (APPROXIMATELY 2/3G ACCELERATION)

Portions of the noisy track and filtered track are shown in Figure 2. While the noise corruption cannot totally be eliminated, the filtered track does provide a truer representation of the actual target path. Further improvement can be achieved by applying current post-run smoothing techniques to this filtered track.

B. MANEUVERING TARGET

For tracking exercises run with a target that is known to be maneuvering, the gain schedule can be adjusted to follow those maneuvers. If, for example, the exercise is to be run with a MK 30 simulator programmed for maneuvers involving accelerations of less than 10 feet/second^2, a gain schedule as listed in Table 2 would be used. The higher gain values needed would reduce the quality of the real-time smoothing.

The degree of smoothing accomplished can be seen in Figure 3, a comparison of portions of the tracks obtained from raw, measured data and data smoothed using the gains derived for a maneuvering target.

The reduction in the quality of smoothing may be seen in Figure 4. These filtered tracks were obtained using the maneuvering and non-maneuvering gain schedules. Both tracks were obtained from the same set of noisy, measured data.

Again, the deviation of the filtered track points about the true track is less than that displayed by the measured positions. Favorable comparison was made between the experimental and theoretical values of the error statistics.

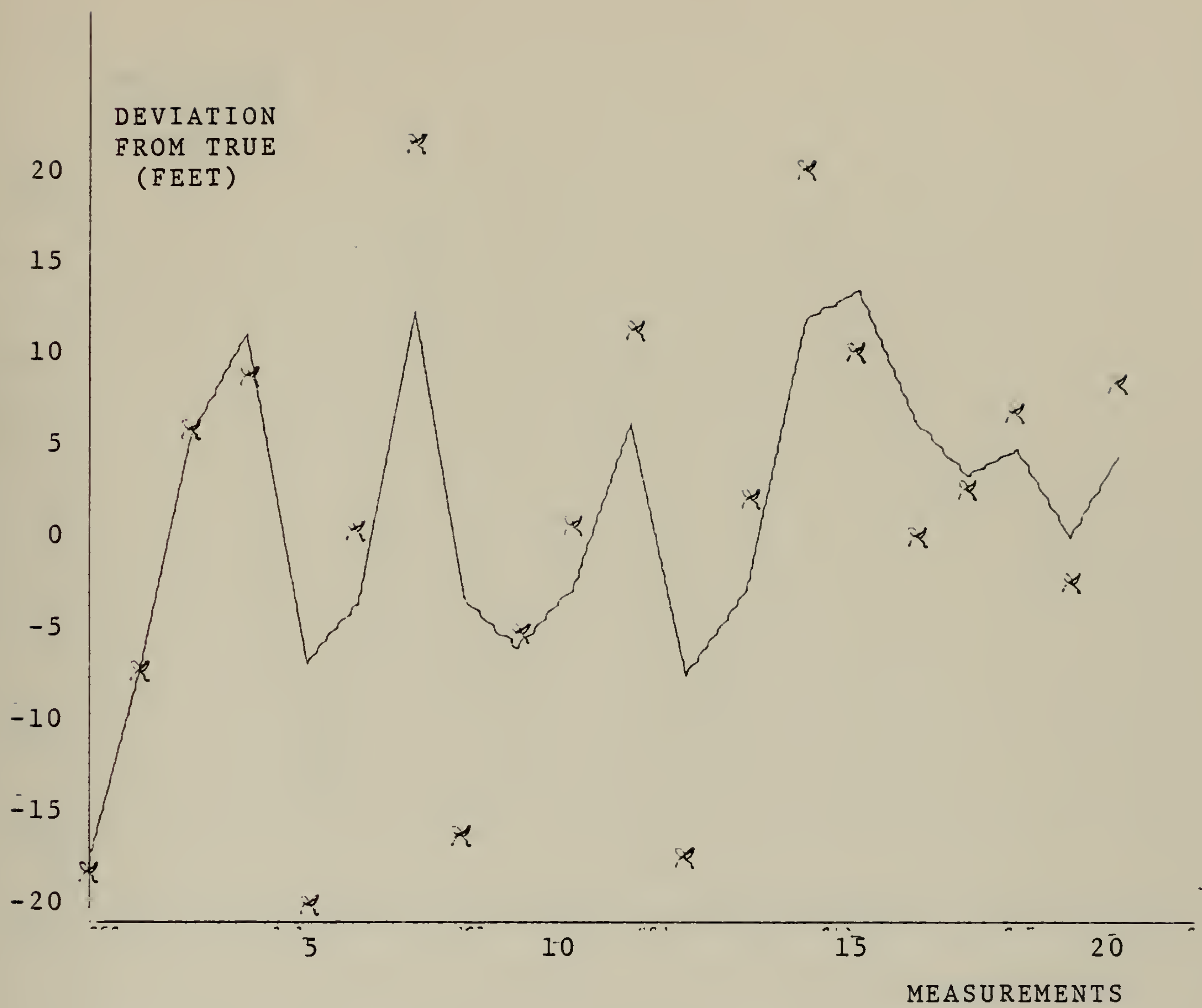

Figure 2 - COMPARISON OF THE NOISY AND FILTERED TRACK POINTS FOR A NON-MANEUVERING TARGET

The gain schedule used to produce the smoothed track above is for a non-maneuvering target. The degree of smoothing attained is seen by comparison of the measured track points (X) and the filtered track points (curve).

The gains listed in Table 1 were used to filter this data. This plot represents the maximum smoothing achieved in the simulation. The magnitude of the Y-axis fluctuation is due to measurement noise only.

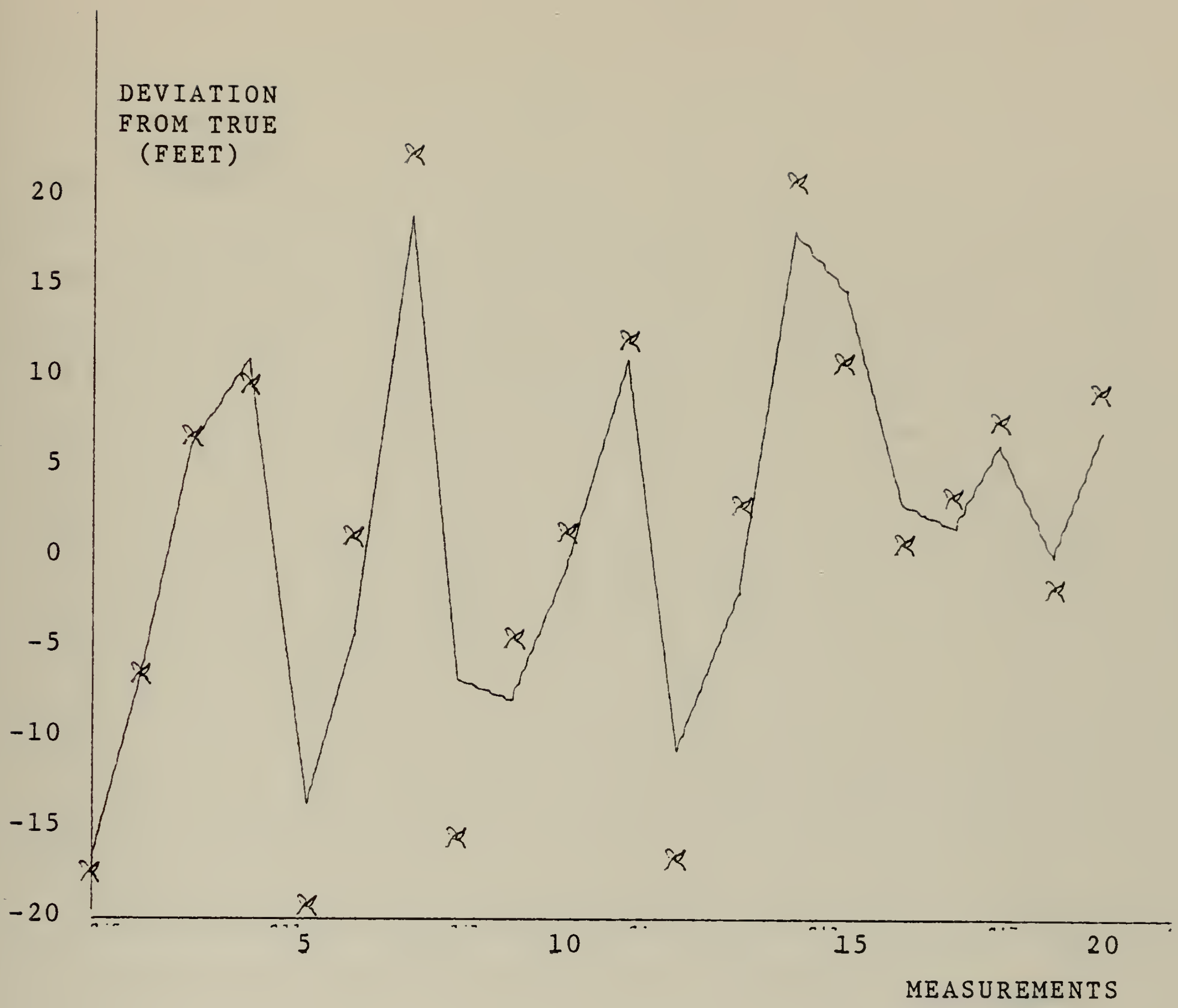

Figure 3 - COMPARISON OF THE NOISY AND FILTERED TRACK POINTS FOR A MANEUVERING TARGET.

The gain schedule used to produce this smoothed track is for a maneuvering target. Again comparison is made between the noisy track measurements (X) and the filtered track points (curve).

The gains listed in Table 2 were used to filter this data. Again the Y-axis fluctuation is due to measurement noise only.

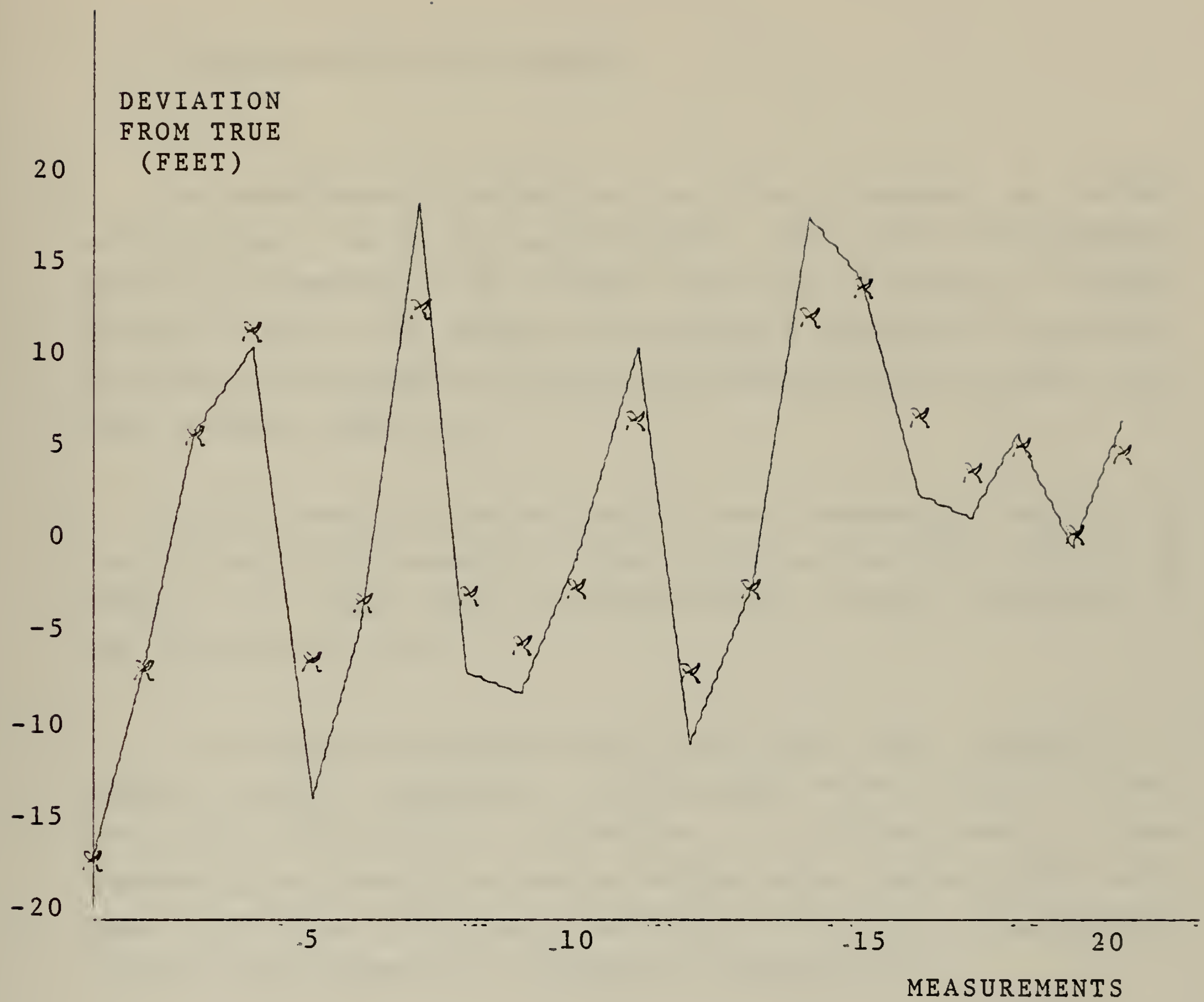

Figure 4 - COMPARISON OF THE FILTERED TRACKS USING THE MANEUVERING AND NON-MANEUVERING GAIN SCHEDULES.

The plot shown above is a comparison of the difference in the degree of smoothing achieved with the gain schedules previously used. The non-maneuvering, filtered track from Figure 2 is displayed with a (X), and the maneuvering filtered track of Figure 3 is shown by the curve.

C. HIGHLY-MANEUVERING TARGET

One additional simulation was run assuming a target accelerating up to 20 feet/second^2. This high acceleration might be expected if an attempt was made to follow a torpedo through both the search and pursuit phases of its track. Allowances for high acceleration substantially increase the gain values (Table 3).

With these high gains most of the smoothing effect of the filter is eliminated. This point is best illustrated in Figure 5, a comparison of measured and filtered portions of the simulated track.

Care must be exercised when the gains are adjusted to follow large maneuvers. If the gate value is set too low, the filter will fail to function. In this test it was necessary to double the value of the gate limits (±100 feet) used in the two previous simulations. (Further explanation of the gate feature is included in Appendix B.)

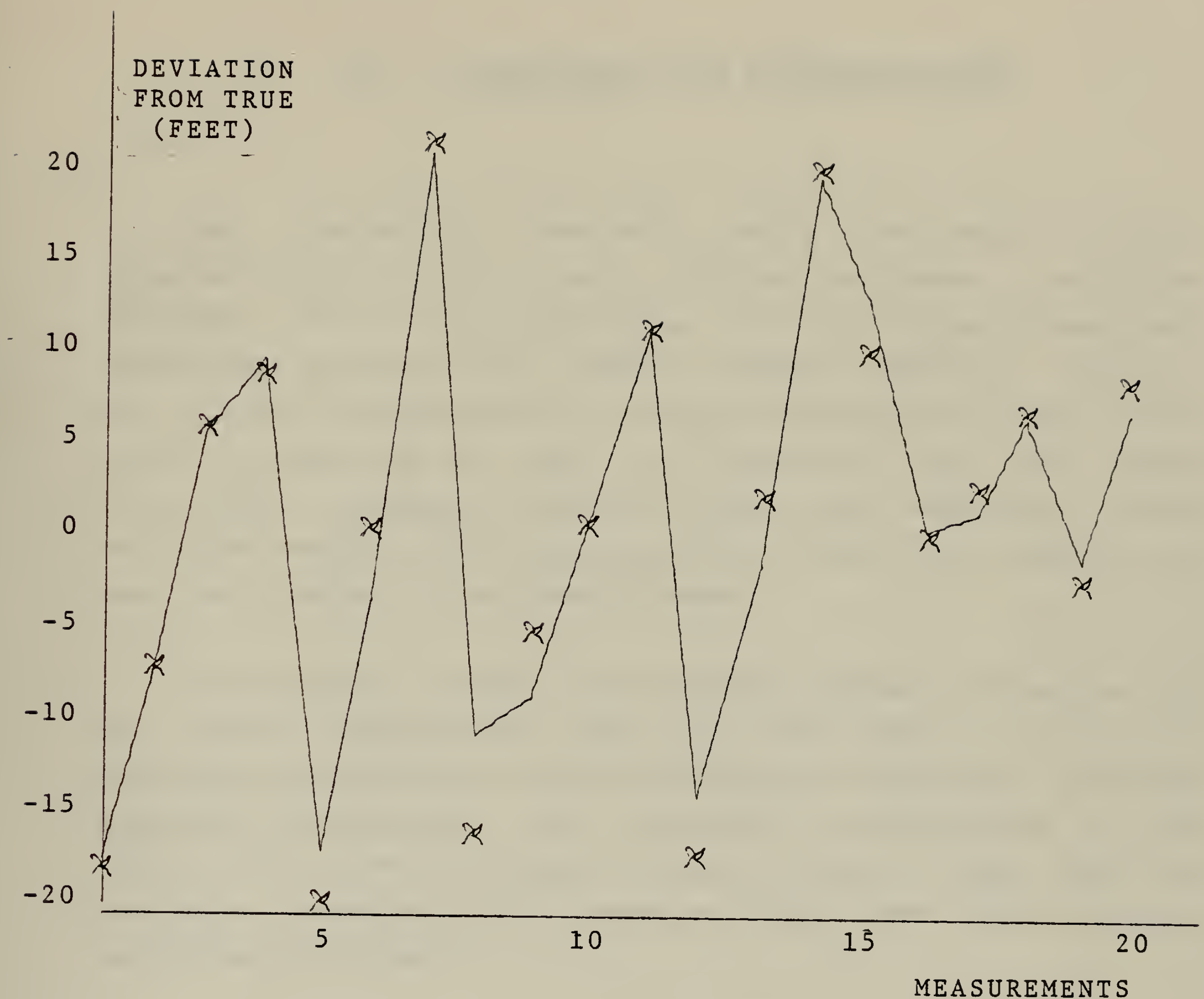

Figure 5 - COMPARISON OF THE NOISY AND FILTERED TRACKS FOR A HIGHLY MANEUVERING TARGET.

The gain schedule used to produce the filtered track above is for a highly maneuvering target. Almost a total lack of smoothing is noted in comparison of the noisy track measurements (X) and the filtered track points (curve).

The gain schedule listed in Table 3 was used in processing this data. The ability to follow large maneuvers results in the lack of smoothing seen.

VII. CONCLUSIONS AND RECOMMENDATIONS

The simulations conducted using the Kalman filter program indicate that it will provide improved, real-time tracking data for operations on the three-dimensional underwater ranges at the Naval Torpedo Station, Keyport. The filter accomplishes a partial smoothing of the target track in real-time as well as providing the long range prediction capability desired by NTS range engineers. Both of these features will substantially improve the quality of service provided to NTS range users.

Additionally, either the filtered position estimates or the actual measurement data may be used in any type of smoothing algorithm to obtain the type of post-run analysis currently available. The smoothing accomplished by the filter will provide a higher quality input to the post-run analysis, resulting in a smoother final track than has been previously attained.

With the exception of minor input/output change requirements, the gain and filter programs are ready for implementation at range computer facilities at NTS. Explanation of these changes is included in the program requirements, Appendices A and B.

Based on the assumption that the search mode of any torpedo tracking exercise is insignificant, the optimal tracking and smoothing can be accomplished using the gain schedule suggested for a non-maneuvering target (Table 1). Before implementation additional testing and comparison may be desirable using exercise data obtained under a variety of

operating conditions at NTS.

In the future, additional improvement to the tracking system may be accomplished by altering the data input format. As stated earlier, the target's measured position expressed in the three-dimensional coordinate system of the range was used as input to the Kalman filter program. Direct application of the time signal to a filter sequence has not been considered at this time. This application may be considered as a possibility for futher program development.

Several references have been made to the smoothing techniques used in current post-run analysis. No attempt has been made to apply such methods to the filter output as simulated. It is recommended that the filter be applied to previously recorded, range-tracking data and the smoothing alogrithm then be applied to this filtered output. Comparison should then be made between the filtered and measured smooth curves.

APPENDIX A

GAIN PROGRAM DESCRIPTION AND REQUIREMENTS

The Fortran program GAINS was designed to compute a Kalman gain schedule based on user selected input matrices. The gain and covariance equations listed in Chapter II were used in the computations.

The user must input the following integer variables: the dimension of the state vector ($N \leq 6$); the dimension of the measurement vector ($M \leq 3$); and, the number of gains to be calculated ($ITIME \leq 200$).

ITIME will normally be set only high enough to insure that the steady-state value of the gains is reached. This value will usually be less than thirteen. If a permanant change is desired, the dimensions of GK and PKK (line 2) should be changed to read "GK(6,3,XXX)" and "PKK(6,6,XXX)" where XXX represents the desired value.

Additional input requirements are the following matrices: PHI, the state transition matrix (dimensions NxN); C, the measurement matrix (MxN); R, the variance of the measurement noise (MxM); PKKM1, the variance of the initial state estimate (NxN); and, Q, the variance of the random forcing input (NxN). All of these matrices should be input in fixed point arithmetic with no single value exceeding ten characters.

As currently written, the program output is limited to

two gains (position and velocity) and the associated variances and covariance. If additional values are desired, it is necessary to change only statements 43 and 45 as noted in the program listing. The actual gain calculations are performed in SUBROUTINE GAIN. Other subroutines are used to perform the necessary algebraic calculations.

The prcgram output will be a gain schedule and theoretical covariance matrix of the specified length (ITIME). These values are output to separate files. The gain output is used as input to the filter program which is currently written to read 13 gain values.

APPENDIX B

KALMAN FILTER PROGRAM DESCRIPTION AND REQUIREMENTS

This program requires input from three sources. Thirteen values of the gain schedule are read via file three. The actual target position measurements are input via file four. The program is designed to terminate when the measurement value of the X position exceeds 10^5 feet (see line 95). Several values must be input by range observers prior to each run. These values are the initial state estimate (six fixed point numbers, each of less than eight characters), the sample interval, long range predictor and measurement gate value (each of fixed point numbers with up to ten characters each).

The long range predictor is scaled in sample intervals, NOT units of time, so care must be exercised to obtain the desired value.

The gate value is used to reject measurements that are obviously erroneous. This value must be set high enough to allow adequate operation of the filter and still perform its function. In the non-maneuvering simulation the value was set at 50 feet with a standard deviation of ten feet in the generated noise.

As written the program output supplies the predicted, measured and filtered position values and filtered velocity estimate. Three statements (124,128,130) output these values to one file with all X values listed first followed

by all Y values and then Z values.

Upon implementation at NTS, all filtered (position and velocity) and measured (position) values may be output by rewritting line 124 and the associated FORMAT statement (#300, line 134). If this change is made lines 127 through 130, inclusive, should be deleted.

The dimension statement (lines 1,2) should be changed so that XKKM1, Z, and XKK will be able to handle the expected number of measurements in any given exercise run. At present they are set at 200 (approximately four minutes with a sample interval of 1.31072 seconds).

The program contains additional comments to facilitate its use.

APPENDIX C

TRACK SIMULATOR PROGRAM DESCRIPTION AND REQUIREMENTS

The simulator program requires no input. The random number generator (SNORM) used to determine target entry point and measurement noise is seeded with two arbitrary values included in the program (lines 5,6). If the seeds are changed they must be odd integer values of eight or fewer characters.

This program generates a 200 point true track (X) and adds random noise to each track point to obtain a noisy measurement (Z). The two tracks are output to separate files.

The program contains additional comments to facilitate its use.

APPENDIX D

GAINS PROGRAM LISTING

```
      DIMENSION PHI(6,6),GAMMA(6,3),C(3,6),CT(6,3),R(3,3),PKKM1(6,6),
     1 EI(6,6),GK(6,3,200),G(6,3),PKK(6,6,200),P(6,6),Q(6,6)
C
      READ(4,200) N,M,ITIME
      CALL MREAD(PHI,N,N)
      CALL MREAD(C,M,N)
      CALL MREAD(R,M,M)
      CALL MREAD(PKKM1,N,N)
      CALL MREAD(Q,N,N)
C
C    SET UP AN IDENTITY MATRIX
      DC 10 I=1,N
      DO 10 J=1,N
 10   EI(I,J)=0.0
      DC 11 I=1,N
 11   EI(I,I)=1.0
C
C    K = 2 IS TIME = 1
C
C
C    ESTABLISH THE GAIN SCHEDULE
      DO 30 K=1,ITIME
      CALL GAIN (PKKM1,P,G,R,EI,PHI,C,N,M,CT,Q)
      DC 21 I=1,N
      DO 21 J=1,N
 21   PKK(I,J,K)=P(I,J)
      DO 22 I=1,N
      DC 22 J=1,M
 22   GK(I,J,K)=G(I,J)
 30   CONTINUE
C
C    OUTPUT THE RESULTS FOR LATER USE
C
C*********************************************************************
C
C     THE PROGRAM OUTPUTS ONLY TWO GAINS AS CURRENTLY WRITTEN.
C     TO GET THE ADDITIONAL GAINS (G32, G42, G53, G63) AND THE
C     ASSOCIATED COVARIANCE MATRIX THE USER SHOULD CHANGE THE
C     NEXT TWO WRITE STATEMENTS TO INCLUDE ALL DESIRED VARIABLES.
C
C*********************************************************************
C
      DC 31 K=1,ITIME
 31   WRITE(1,100)GK(1,1,K),GK(2,1,K)
      DO 32 K=1,ITIME
 32   WRITE(2,300)PKK(1,1,K),PKK(1,2,K),PKK(2,2,K)
      STOP
 100  FORMAT(8F10.6)
 200  FCRMAT(3I5)
```

```
300   FORMAT(8F10.2)
      END
C
      SUBROUTINE MREAD (A,N,M)
      DIMENSION A(N,M)
      DO 10 I=1,N
10    READ (4,11) (A(I,J),J=1,M)
11    FCRMAT (8F10.5)
      RETURN
      END
C
      SUBROUTINE PROD (A,B,N,M,L,C)
      DIMENSION A(N,M),B(M,L),C(N,L)
      DC 1 I=1,N
      DO 1 J=1,L
1     C(I,J) = 0.
      DO 2 I=1,N
      DO 2 J=1,L
      DC 2 K=1,M
2     C(I,J) = C(I,J) + A(I,K) * B(K,J)
      RETURN
      END
C
      SUBROUTINE SUB (A,B,N,M,C)
      DIMENSION A(N,M),B(N,M),C(N,M)
      DO 1 I=1,N
      DO 1 J=1,M
1     C(I,J) = A(I,J) - B(I,J)
      RETURN
      END
C
      SUBROUTINE ADD (A,B,N,M,C)
      DIMENSION A(N,M),B(N,M),C(N,M)
      DC 1 I=1,N
      DO 1 J=1,M
1     C(I,J) = A(I,J) + B(I,J)
      RETURN
      END
C
      SUBROUTINE TRANS (A,N,M,B)
      DIMENSION A(N,M),B(M,N)
      DC 1 I=1,N
      DO 1 J=1,M
1     B(J,I) = A(I,J)
      RETURN
      END
C
      SUBROUTINE GAIN (PKKM1,PKK,G,R,EI,PHI,C,N,M,CT,Q)
```

```
      DIMENSION PKKM1(6,6),PKK(6,6),G(6,3),R(3,3),PHI(6,6),C(3,6),      NTS0097
     * CT(6,3),TEMP(6,6),TEMP1(3,3),TEMP2(6,6),TEMP3(3,3),EI(6,6),Q(6,6)NTS0098
      CALL TRANS (C,M,N,CT)                                             NTS0099
      CALL PROD (PKKM1,CT,N,N,M,TEMP)                                   NTS0100
      CALL PROD (C,TEMP,M,N,M,TEMP1)                                    NTS0101
      CALL ADD(TEMP1,R,M,M,TEMP1)                                       NTS0102
      CALL RECIP(3,0.0000001,TEMP1,TEMP3,KER,3)                         NTS0103
      CALL PROD(TEMP,TEMP3,N,M,M,G)                                     NTS0104
      CALL PROD (G,C,N,M,N,TEMP)                                        NTS0105
      CALL SUB (EI,TEMP,N,N,TEMP2)                                      NTS0106
      CALL PROD (TEMP2,PKKM1,N,N,N,PKK)                                 NTS0107
      CALL TRANS (PHI,N,N,TEMP)                                         NTS0108
      CALL PROD (PKK,TEMP,N,N,N,TEMP2)                                  NTS0109
      CALL PROD (PHI,TEMP2,N,N,N,TEMP)                                  NTS0110
      CALL ADD(TEMP,Q,N,N,PKKM1)                                        NTS0111
      RETURN                                                            NTS0112
      END                                                               NTS0113
C                                                                       NTS0114
      SUBROUTINE CONST(Q,A,N,M,C,ND,MD)                                 NTS0115
      DIMENSION A(ND,MD),C(ND,MD)                                       NTS0116
      IF(Q)11,10,11                                                     NTS0117
   10 DO 100 I=1,N                                                      NTS0118
      DO 100 J=1,M                                                      NTS0119
  100 C(I,J) = 0.0                                                      NTS0120
      RETURN                                                            NTS0121
   11 IF(Q-1.0)13,12,13                                                 NTS0122
   12 DO 120 I=1,N                                                      NTS0123
      DO 120 J=1,M                                                      NTS0124
  120 C(I,J) = A(I,J)                                                   NTS0125
      RETURN                                                            NTS0126
   13 IF(Q+1.0)15,14,15                                                 NTS0127
   14 DO 140 I=1,N                                                      NTS0128
      DO 140 J=1,M                                                      NTS0129
  140 C(I,J) = -A(I,J)                                                  NTS0130
      RETURN                                                            NTS0131
   15 DO 150 I=1,N                                                      NTS0132
      DO 150 J=1,M                                                      NTS0133
  150 C(I,J) = Q*A(I,J)                                                 NTS0134
      RETURN                                                            NTS0135
      END                                                               NTS0136
C                                                                       NTS0137
      SUBROUTINE   RECIP(N,EP,B,X,KER,M)                                NTS0138
      DIMENSION A(3,3),X(M,M),B(M,M)                                    NTS0139
      CALL CONST(1.,B,N,N,A,3,3)                                        NTS0140
      DO 1 J=1,M                                                        NTS0141
      DC 1 I=1,M                                                        NTS0142
    1 X(I,J)=0.                                                         NTS0143
      DO 2 K=1,N                                                        NTS0144
```

```
  2 X(K,K)=1.                                          NTS0145
 10 DO 34 L=1,N                                        NTS0146
    KP=0                                               NTS0147
    Z=0.                                               NTS0148
    DO 12 K=L,N                                        NTS0149
    IF(Z.GE.ABS(A(K,L))) GO TO 12                      NTS0150
 11 Z=ABS(A(K,L))                                      NTS0151
    KP=K                                               NTS0152
 12 CCNTINUE                                           NTS0153
    IF(L.GE.KP) GO TO 20                               NTS0154
 13 DO 14 J=L,N                                        NTS0155
    Z=A(L,J)                                           NTS0156
    A(L,J)=A(KP,J)                                     NTS0157
 14 A(KP,J)=Z                                          NTS0158
    DO 15 J=1,N                                        NTS0159
    Z=X(L,J)                                           NTS0160
    X(L,J)=X(KP,J)                                     NTS0161
 15 X(KP,J)=Z                                          NTS0162
 20 IF(ABS(A(L,L)).LE.EP) GO TO 50                     NTS0163
 30 IF(L.GE.N) GO TO 34                                NTS0164
 31 LP1=L+1                                            NTS0165
    DO 36 K=LP1,N                                      NTS0166
    IF(A(K,L).EQ.0.) GO TO 36                          NTS0167
 32 RATIO=A(K,L)/A(L,L)                                NTS0168
    DO 33 J=LP1,N                                      NTS0169
 33 A(K,J)=A(K,J)-RATIO*A(L,J)                         NTS0170
    DO 35 J=1,N                                        NTS0171
 35 X(K,J)=X(K,J)-RATIO*X(L,J)                         NTS0172
 36 CONTINUE                                           NTS0173
 34 CCNTINUE                                           NTS0174
 40 DC 43 I=1,N                                        NTS0175
    I1=N+1-I                                           NTS0176
    DO 43 J=1,N                                        NTS0177
    S=0.                                               NTS0178
    IF(I1.GE.N) GO TO 43                               NTS0179
 41 IIP1=I1+1                                          NTS0180
    DO 42 K=IIP1,N                                     NTS0181
 42 S=S+A(I1,K)*X(K,J)                                 NTS0182
 43 X(I1,J)=(X(I1,J)-S)/A(I1,I1)                       NTS0183
    KER=1                                              NTS0184
    RETURN                                             NTS0185
 50 KER=2                                              NTS0186
    RETURN                                             NTS0187
    END                                                NTS0188
```

APPENDIX E

KALMAN FILTER PROGRAM LISTING

```
      DIMENSION XKKM1(6,200),XKK(6,200),GK(6,3,200),Z(3,200),
     1 ERROR(3,200)
C
      READ(5,100)(XKKM1(I,1),I=1,6)
      READ(5,200)T,LONG,GATE
      READ(4,300)(Z(I,1),I=1,3)
      READ(3,400)GK(1,1,1),GK(2,1,1)
      ERROR(1,1) = Z(1,1) - XKKM1(1,1)
      ERROR(2,1) = Z(2,1) - XKKM1(3,1)
      ERROR(3,1) = Z(3,1) - XKKM1(5,1)
      XKK(1,1) = XKKM1(1,1) + GK(1,1,1) * ERROR(1,1)
      XKK(2,1) = XKKM1(2,1) + GK(2,1,1) * ERROR(1,1)
      XKK(3,1) = XKKM1(3,1) + GK(1,1,1) * ERROR(2,1)
      XKK(4,1) = XKKM1(4,1) + GK(2,1,1) * ERROR(2,1)
      XKK(5,1) = XKKM1(5,1) + GK(1,1,1) * ERROR(3,1)
      IF (XKK(5,1).GT.0.0) XKK(5,1) = 0.0
C
C     THE ABOVE STATEMENT PREVENTS POSITIONING OF THE TARGET ABOVE
C     THE WATER DUE TO A NOISY MEASUREMENT.
C
      XKK(6,1) = XKKM1(6,1) + GK(2,1,1) * ERROR(3,1)
C
      WRITE(10,300)XKKM1(1,1),Z(1,1),XKK(1,1),XKK(2,1)
C
C     THE FCLLOWING LOOP OPERATES THE FILTER FOR 12 INTERVALS UNTIL
C     THE STEADY STATE GAIN VALUES ARE REACHED
C
      DO 30 K=2,12
C
C     PREDICT THE NEXT POINT
C
      XKKM1(1,K) = XKK(1,K-1) + T * XKK(2,K-1)
      XKKM1(2,K) = XKK(2,K-1)
      XKKM1(3,K) = XKK(3,K-1) + T * XKK(4,K-1)
      XKKM1(4,K) = XKK(4,K-1)
      XKKM1(5,K) = XKK(5,K-1) + T * XKK(6,K-1)
      XKKM1(6,K) = XKK(6,K-1)
C
C     TAKE THE NEXT MEASUREMENT
C
      READ(4,300)(Z(I,K),I=1,3)
C
C     DETERMINE THE ERROR FOR THIS TRACK POINT
C
      ERROR(1,K) = Z(1,K) - XKKM1(1,K)
      ERROR(2,K) = Z(2,K) - XKKM1(3,K)
      ERROR(3,K) = Z(3,K) - XKKM1(5,K)
C
```

```
C    UPDATE THE ESTIMATE  X(K/K) = X(K/K-1) - G * (Z - C * X(K/K-1))
C
      READ(3,400)GK(1,1,K),GK(2,1,K)
      IF(ABS(ERROR(1,K)).GT.GATE) XKK(1,K) = XKKM1(1,K)
      IF(ABS(ERROR(1,K)).GT.GATE) XKK(2,K) = XKKM1(2,K)
      IF(ABS(ERROR(1,K)).GT.GATE) GO TO 10
      XKK(1,K) = XKKM1(1,K) + GK(1,1,K) * ERROR(1,K)
      XKK(2,K) = XKKM1(2,K) + GK(2,1,K) * ERROR(1,K)
 10   IF(ABS(ERROR(2,K)).GT.GATE) XKK(3,K) = XKKM1(3,K)
      IF(ABS(ERROR(2,K)).GT.GATE) XKK(4,K) = XKKM1(4,K)
      IF(ABS(ERROR(2,K)).GT.GATE) GO TO 11
      XKK(3,K) = XKKM1(3,K) + GK(1,1,K) * ERROR(2,K)
      XKK(4,K) = XKKM1(4,K) + GK(2,1,K) * ERROR(2,K)
 11   IF(ABS(ERROR(3,K)).GT.GATE) XKK(5,K) = XKKM1(5,K)
      IF(ABS(ERROR(3,K)).GT.GATE) XKK(6,K) = XKKM1(6,K)
      IF(ABS(ERROR(3,K)).GT.GATE) GO TO 12
      XKK(5,K) = XKKM1(5,K) + GK(1,1,K) * ERROR(3,K)
      IF (XKK(5,K).GT.0.0) XKK(5,K) = 0.0
C
C    THE ABOVE STATEMENT PREVENTS POSITIONING OF THE TARGET ABOVE
C    THE WATER DUE TO A NOISY MEASUREMENT.
C
      XKK(6,K) = XKKM1(6,K) + GK(2,1,K) * ERROR(3,K)
 12   WRITE(10,300)XKKM1(1,K),Z(1,K),XKK(1,K),XKK(2,K)
      XLONG = XKK(1,K) + LONG*T*XKK(2,K)
      YLONG = XKK(3,K) + LONG*T*XKK(4,K)
      ZLONG = XKK(5,K) + LONG*T*XKK(6,K)
 30   CONTINUE
      K=13
C
C    CONTINUE THE FILTER OPERATION WITH GAINS AT STEADY STATE VALUES
C
      READ(3,400)GSSP,GSSV
C
C    PREDICT THE NEXT POINT
C
 31   XKKM1(1,K) = XKK(1,K-1) + T * XKK(2,K-1)
      XKKM1(2,K) = XKK(2,K-1)
      XKKM1(3,K) = XKK(3,K-1) + T * XKK(4,K-1)
      XKKM1(4,K) = XKK(4,K-1)
      XKKM1(5,K) = XKK(5,K-1) + T * XKK(6,K-1)
      XKKM1(6,K) = XKK(6,K-1)
C
C    TAKE THE NEXT MEASUREMENT
C
      READ(4,300)Z(1,K),Z(2,K),Z(3,K)
      IF(Z(1,K).GT.99999.) GO TO 35
C
```

```
C     DETERMINE THE ERROR FOR THIS TRACK POINT
C
      ERROR(1,K) = Z(1,K) - XKKM1(1,K)
      ERROR(2,K) = Z(2,K) - XKKM1(3,K)
      ERROR(3,K) = Z(3,K) - XKKM1(5,K)
C
C     UPDATE THE ESTIMATE
C
      IF(ABS(ERROR(1,K)).GT.GATE) XKK(1,K) = XKKM1(1,K)
      IF(ABS(ERROR(1,K)).GT.GATE) XKK(2,K) = XKKM1(2,K)
      IF(ABS(ERROR(1,K)).GT.GATE) GO TO 32
      XKK(1,K) = XKKM1(1,K) + GSSP * ERROR(1,K)
      XKK(2,K) = XKKM1(2,K) + GSSV * ERROR(1,K)
32    IF(ABS(ERROR(2,K)).GT.GATE) XKK(3,K) = XKKM1(3,K)
      IF(ABS(ERROR(2,K)).GT.GATE) XKK(4,K) = XKKM1(4,K)
      IF(ABS(ERROR(2,K)).GT.GATE) GO TO 33
      XKK(3,K) = XKKM1(3,K) + GSSP * ERROR(2,K)
      XKK(4,K) = XKKM1(4,K) + GSSV * ERROR(2,K)
33    IF(ABS(ERROR(3,K)).GT.GATE) XKK(5,K) = XKKM1(5,K)
      IF(ABS(ERROR(3,K)).GT.GATE) XKK(6,K) = XKKM1(6,K)
      IF(ABS(ERROR(3,K)).GT.GATE) GO TO 34
      XKK(5,K) = XKKM1(5,K) + GSSP * ERROR(3,K)
      IF(XKK(5,K).GT.0.0) XKK(5,K) = 0.0
      XKK(6,K) = XKKM1(6,K) + GSSV * ERROR(3,K)
34    XLONG = XKK(1,K) + LONG * T * XKK(2,K)
      YLONG = XKK(3,K) + LONG * T * XKK(4,K)
      ZLONG = XKK(5,K) + LONG * T * XKK(6,K)
      K = K+1
      GO TO 31
35    DO 40 K=1,100
40    WRITE(10,300)XKKM1(3,K),Z(2,K),XKK(3,K),XKK(4,K)
      DO 50 K=1,100
50    WRITE(10,300)XKKM1(5,K),Z(3,K),XKK(5,K),XKK(6,K)
      STOP
100   FORMAT(8F7.2)
200   FORMAT(3F10.8)
300   FORMAT(8F10.2)
400   FCRMAT(8F10.6)
500   FCRMAT(2(2X,I10))
      END
```

APPENDIX F

TRACK SIMULATOR PROGRAM LISTING

```
      DIMENSICN X(6,200),Z(3,200),OUT(3),V(3)
C
C    IX AND IV ARE THE INITIAL SNORM "SEEDS"
C
      IX=12649
      IV=101143
C
C    ASSUME WATER ENTRY POINT IS RANDOM ON (0,50)
C
      CALL SNORM(IX,OUT,3)
      X(1,1) = OUT(1) * 50.
      X(3,1) = OUT(2) * 50.
      X(5,1) = OUT(3) * 50. - 100.
      CALL SNORM(IV,V,3)
      Z(1,1) = X(1,1) + V(1) * 10.
      Z(2,1) = X(3,1) + V(2) * 10.
      Z(3,1) = X(5,1) + V(3) * 10.
      WRITE(1,20)X(1,1),X(3,1),X(5,1)
      WRITE(2,20)Z(1,1),Z(2,1),Z(3,1)
C
C    GENERATE REMAINDER OF TRACK AND ADD NOISE TO MEASUREMENTS
C
      DC 10 N=2,100
C
C    NOTE X VELOCITY ONLY AT  40 KNOTS
C
      X(1,N) = X(1,N-1) + 90.
      X(3,N) = X(3,N-1)
      X(5,N) = X(5,N-1)
C
C    ADD NOISE TO OBTAIN "MEASUREMENTS"
C
      CALL SNORM (IV,V,3)
      Z(1,N) = X(1,N) + V(1) * 10.
      Z(2,N) = X(3,N) + V(2) * 10.
      Z(3,N) = X(5,N) + V(3) * 10.
C
C    TRUE AND NOISY TRACKS ARE OUTPUT TO DIFFERENT FILES FOR EASE
C    CF LATER USE
C
      WRITE(1,20)X(1,N),X(3,N),X(5,N)
   10 WRITE(2,20)Z(1,N),Z(2,N),Z(3,N)
   20 FORMAT(8F10.2)
      STOP
      END
```

LIST OF REFERENCES

1. Gelb, Arthur, Applied Optimal Estimation, p. 102-142, M.I.T. Press, 1974.

2. Kirk, Donald E., EE4414 Class Notes (unpublished), Naval Postgraduate School, 1975.

3. NAVORD OD 41114, Data Reduction Requirements, 3-D Data, 15 February 1973.

4. NAVAL TORPEDO STATION REPORT 1049, Validity of the Centerline Coordinate System at the Nanoose Range, by J. F. Ebert, 10 December 1969.

5. NAVORD OD 41115, Description and Operation, 3-D Range Equipment, 15 February 1973.

INITIAL DISTRIBUTION LIST

		No. Copies
1.	Defense Documentation Center Camden Station Arlington, Virginia 22314	2
2.	Library (Code 0212) Naval Postgraduate School Monterey, California 93940	2
3.	Department Chairman, Code 62 Department of Electrical Engineering Naval Postgraduate School Monterey, California 93940	2
4.	Professor Harold A. Titus, Code 62Ts Department of Electrical Engineering Naval Postgraduate School Monterey, California 93940	10
5.	Commanding Officer Naval Torpedo Station Keyport, Washington 98345	2
6.	Lt. Eric James Benson, USN 735 Huntington Court Martinez, California 94553	2

168893
Thesis
B3915 Benson
c.1 An application of
Kalman filtering to
underwater tracking.